Package ‘multipleOutcomes’

May 18, 2026

Title Joint Covariance and Treatment-Effect Tests for Multiple Outcomes

Version 0.16.2

Description Fits generalized linear models, Cox proportional-hazards models, log-rank tests, generalized estimating equations, mixed models with repeated measures, Kaplan-Meier curves, and quantile differences jointly across multiple endpoints, and returns the full asymptotic covariance matrix linking them. Implements PATED (Prognostic Assisted Treatment Effect Detection), a randomized-trial method that exploits balanced prognostic covariates to tighten standard errors and increase statistical power without introducing bias.

URL `https://github.com/zhangh12/multipleOutcomes`, `https://zhangh12.github.io/multipleOutcomes/`

BugReports `https://github.com/zhangh12/multipleOutcomes/issues`

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports dplyr, ggplot2, ggpubr, mmrm (>= 0.3.15), mvtnorm, rlang, sandwich, stringr, survival, tidyr, tidyselect

Suggests asaur, coin, iBST, invGauss, JM, joint.Cox, knitr, momentfit, numDeriv, pec, randomForestSRC, rmarkdown, testthat (>= 3.0.0)

Depends R (>= 3.5)

LazyData true

NeedsCompilation yes

Author Han Zhang [aut, cre]

Maintainer Han Zhang <`zhangh.ustc@gmail.com`>

Repository CRAN

Date/Publication 2026-05-18 07:00:09 UTC

Contents

`actg` *ACTG 320 Clinical Trial Dataset*

Description

actg dataset from Hosmer et al.

Format

A data frame

id Identification Code

time Time to AIDS diagnosis or death (days).

censor Event indicator. 1 = AIDS defining diagnosis, 0 = Otherwise.

time_d Time to death (days)

censor_d Event indicator for death (only). 1 = Death, 0 = Otherwise.

tx Treatment indicator. 1 = Treatment includes IDV, 0 = Control group.

txgrp Treatment group indicator. 1 = ZDV + 3TC. 2 = ZDV + 3TC + IDV. 3 = d4T + 3TC. 4 = d4T + 3TC + IDV.

strat2 CD4 stratum at screening. 0 = CD4 <= 50. 1 = CD4 > 50.

sex 0 = Male. 1 = Female.

raceth Race/Ethnicity. 1 = White Non-Hispanic. 2 = Black Non-Hispanic. 3 = Hispanic. 4 = Asian, Pacific Islander. 5 = American Indian, Alaskan Native. 6 = Other/unknown.

ivdrug IV drug use history. 1 = Never. 2 = Currently. 3 = Previously.

hemophil Hemophiliac. 1 = Yes. 0 = No.

karnof Karnofsky Performance Scale. 100 = Normal; no complaint no evidence of disease. 90 = Normal activity possible; minor signs/symptoms of disease. 80 = Normal activity with effort; some signs/symptoms of disease. 70 = Cares for self; normal activity/active work not possible.

cd4 Baseline CD4 count (Cells/Milliliter).

priorzdv Months of prior ZDV use (months).

age Age at Enrollment (years).

Source

ftp://ftp.wiley.com/public/sci_tech_med/survival

References

Hosmer, D.W. and Lemeshow, S. and May, S. (2008) Applied Survival Analysis: Regression Modeling of Time to Event Data: Second Edition, John Wiley and Sons Inc., New York, NY

Examples

```
data(actg)
```

`coef.jointCovariance` *Extract Model Coefficients*

Description

`coef` is a generic function.

Usage

```
## S3 method for class 'jointCovariance'
coef(object, model_index = NULL, ...)
```

Arguments

`object`	an object returned by `jointCovariance()`.
`model_index`	`NULL` if displaying coefficients of all fitted models; otherwise, an integer indicating the fitted model.
`...`	for debugging only

Value

a vector of coefficient estimates

`coxph_` *Creating Objects of Proportional Hazards Regression Model*

Description

`coxph_` is a wrapper function of `survival::coxph` to create an object to be passed into `jointCovariance`, the main function of this package through its argument `...`. The object defines how a proportional hazard model would be fitted.

Usage

```
coxph_(formula, data_index = 1)
```

Arguments

`formula`	see `formula` in `survival::coxph`.
`data_index`	integer. Index of the data frame in the `data` argument of `jointCovariance` to be used when fitting a proportional hazards model.

Details

Not all arguments of `survival::coxph` are supported in `coxph_` due to the complexity in handling environment and scope, which is particularly difficult for arguments like `weights`, `subset`, etc.

`gee_` *Creating Objects of Generalized Estimation Equation Model*

Description

`gee_` is a wrapper function of `gee::gee` to create an object to be passed into `jointCovariance`, the main function of this package through its argument `...`. The object defines how a GEE model would be fitted.

This package does not import the package gee. Instead, codes of `gee` are modified and integrated to compute score and information matrix. Thus, users does not need to install the package gee to use this package.

Usage

```
gee_(formula, family, corstr, R = NULL, b = NULL, Mv = 1, data_index = 1)
```

Arguments

`formula`	see `formula` in `gee::gee`.
`family`	see `family` in `gee::gee`.
`corstr`	see `corstr` in `gee::gee`.
`R`	see `R` in `gee::gee`.
`b`	see `b` in `gee::gee`.
`Mv`	see `Mv` in `gee::gee`.
`data_index`	integer. Index of the data frame in the `data` argument of `jointCovariance` to be used when fitting a GEE model.

Details

Not all arguments of `stats::gee` are supported in `gee_` due to the complexity in handling environment and scope, which is particularly difficult for arguments like `subset`, etc.

`glm_` *Creating Objects of Generalized Linear Models*

Description

`glm_` is a wrapper function of `stats::glm` to create an object to be passed into `jointCovariance`, the main function of this package through its argument `...`. The object defines how a GLM model would be fitted.

Usage

```
glm_(formula, family, data_index = 1)
```

Arguments

`formula`	see `formula` in `stats::glm`.
`family`	currently supports `"gaussian"` or `"binomial"`. Other families are under testing.
`data_index`	integer. Index of the data frame in the `data` argument of `jointCovariance` to be used when fitting a generalized linear model.

Details

Not all arguments of `stats::glm` are supported in `glm_` due to the complexity in handling environment and scope, which is particularly difficult for arguments like `weights`, `subset`, etc.

`indo` *Rectal Indomethacin for Prevention of Post-ERCP Pancreatitis*

Description

Patient-level data from a multi-center, randomized, double-blind, placebo-controlled 2-arm trial (n = 602) of rectal indomethacin (100 mg) versus placebo to prevent post-ERCP pancreatitis in high-risk patients, as reported by Elmunzer, Higgins, et al. (2012) in the *New England Journal of Medicine*.

This dataset was originally collected, cleaned, reformatted, and released for public teaching and research use by Dr. Peter D. R. Higgins in the **medicaldata** R package as `indo_rct`. The version shipped here is redistributed in support of the worked examples in this package. The variable definitions below follow **medicaldata**; note that a few columns are stored as numeric (rather than factor) in this copy. Users who need the authoritative copy or accompanying documentation should consult **medicaldata**.

Format

A data frame with 602 observations on the following 33 variables:

`id` Subject identifier (numeric); leading digit indicates center. Range 1001–4003.

`site` Study site (factor, 4 levels): `1_UM` = University of Michigan, `2_IU` = Indiana University, `3_UK` = University of Kentucky, `4_Case` = Case Western Reserve University.

`age` Age in years (numeric), range 19–90.

`risk` Risk score for post-ERCP pancreatitis (numeric), range 1–5.5.

`gender` Sex (factor): `1_female`, `2_male`.

`outcome` Primary outcome: post-ERCP pancreatitis (numeric, 1 = yes, 0 = no).

`sod` Sphincter of Oddi dysfunction present (factor): `0_no`, `1_yes`.

`pep` History of prior post-ERCP pancreatitis (factor): `0_no`, `1_yes`.

`recpanc` History of recurrent pancreatitis (factor): `0_no`, `1_yes`.

`psphinc` Pancreatic sphincterotomy performed (factor): `0_no`, `1_yes`.

`precut` Sphincter pre-cut needed to enter papilla (factor): `0_no`, `1_yes`.

`difcan` Cannulation of papilla was difficult (factor): `0_no`, `1_yes`.

`pneudil` Pneumatic dilation of papilla performed (factor): `0_no`, `1_yes`.

`amp` Ampullectomy performed (factor): `0_no`, `1_yes`.

`paninj` Contrast injected into pancreas (factor): `0_no`, `1_yes`.

`acinar` Pancreas appeared to have acinarization on imaging (factor): `0_no`, `1_yes`.

`brush` Brushings taken from pancreatic duct (factor): `0_no`, `1_yes`.

`asa81` Aspirin used at 81 mg per day (factor with 3 levels): `0_no`, `1_yes`, and a third level retained from the source coding.

`asa325` Aspirin used at 325 mg per day (factor with 3 levels): `0_no`, `1_yes`, and a third level retained from the source coding.

`asa` Aspirin used at any dose (factor with 3 levels): `0_no`, `1_yes`, and a third level retained from the source coding.

`prophystent` Pancreatic duct stent placed per endoscopist judgment (factor): `0_no`, `1_yes`.

`therastent` Pancreatic duct stent placed to treat narrowing (factor): `0_no`, `1_yes`.

`pdstent` Pancreatic duct stent placed for any reason (factor): `0_no`, `1_yes`.

`sodsom` Sphincter of Oddi manometry performed (factor): `0_no`, `1_yes`.

`bsphinc` Biliary sphincterotomy performed (factor): `0_no`, `1_yes`.

`bstent` Biliary stent placed to relieve obstruction (factor): `0_no`, `1_yes`.

`chole` Choledocholithiasis present (factor): `0_no`, `1_yes`.

`pbmal` Biliary duct or pancreatic malignancy found (factor): `0_no`, `1_yes`.

`train` Trainee participated in ERCP (factor): `0_no`, `1_yes`.

`status` Patient status (factor): `0_inpatient`, `1_outpatient`.

`type` Sphincter of Oddi dysfunction type (factor): `0_no SOD`, `1_type 1`, `2_type 2`, `3_type 3`.

`rx` Treatment assignment (numeric, 1 = indomethacin, 0 = placebo).

`bleed` Reportable gastrointestinal bleeding (numeric, coded 1 = no, 2 = yes; `NA` when not assessed).

Source

Higgins, P. D. R. **medicaldata**: Data Package for Medical Datasets. R package, dataset `indo_rct`. `https://CRAN.R-project.org/package=medicaldata`

References

Elmunzer BJ, Higgins PDR, Saini SD, et al. A randomized trial of rectal indomethacin to prevent post-ERCP pancreatitis. *New England Journal of Medicine* 2012; 366(15):1414–1422. doi:10.1056/NEJMoa1111103

Examples

```
data(indo)
```

`jointCovariance` *Fitting Regression Models for Multiple Outcomes and Returning the Matrix of Covariance*

Description

`jointCovariance` can fit different types of models for multiple outcomes simultaneously and return model parameters and variance-covariance matrix for further analysis.

Usage

```
jointCovariance(..., data, nboot = 0, compute_cov = TRUE, seed = NULL)
```

Arguments

`...`	objects returned by `glm_()`, `coxph_()`, `logrank_()`, `gee_()` and `gmm_()`.
`data`	a data frame if all models are fitted on the same dataset; otherwise a list of data frames for fitting models in `...`. Note that a dataset can be used to fit multiple models, thus, `length(data)` is unnecessary to be equal to the number of models in `...`. The row names in a data frame are not treated as subject IDs. Instead, all data frame should consist of a column `pid` as subject IDs. For any two records in different data frames that correspond to the same subject, their values in `pid` should be consistent. For data frames to be used by GEE, `pid` defines clusters. See `id` and the Details section of `gee:gee`.
`nboot`	non-zero integer if bootstrap is adopted. By default 0.
`compute_cov`	logic. If `TRUE` and `nboot > 0`, empirical covariance matrix is computed using bootstrap estimate and returned. Bootstrap estimate will be abandoned. If `FALSE`, bootstrap estimate will be returned and no empirical covariance matrix is computed.
`seed`	random seed when generate bootstrap data.

Value

It returns an object of class "jointCovariance", which is a list containing the following components:

`coefficients`	an unnamed vector of coefficients of all fitted models. Use `id_map` for variable mapping.
`mcov`	a unnamed matrix of covariance of `coefficients`. Use `id_map` for variable mapping.
`id_map`	a list mapping the elements in `coefficients` and `mcov` to variable names.
`n_shared_sample_sizes`	a matrix of shared sample sizes between datasets being used to fit the models.
`call`	the matched call.

Examples

```
## More examples can be found in the vignettes.
library(survival)
library(mvtnorm)
library(tidyr)
genData <- function(seed = NULL){

  set.seed(seed)
  n <- 300
  sigma <- matrix(.7, 4, 4)
  diag(sigma) <- 1
```

```
  v <- rmvnorm(n, sigma = sigma)
  x1 <- v[, 1]
  x2 <- v[, 2]
  z1 <- (v[, 3] > 0) + 0
  z2 <- v[, 4]

  trt <- rbinom(n, 1, .5)

  bet <- c(-.3,.3)
  y <- -log(runif(n))/
    exp(-.3 * x1 + .3 * x2 + z1 * .5 - z2 * .3 + .1 * trt + rnorm(n))

  z1[sample.int(n, 50)] <- NA
  z2[sample.int(n, 50)] <- NA
  x1[sample.int(n, 50)] <- NA
  x2[sample.int(n, 50)] <- NA
  death <- ifelse(y > 2, 0, 1)
  y[y > 2] <- 2

  pid <- paste0('pid-', 1:n)
  ret <- data.frame(
    y = y, trt = trt,
    z1 = z1, z2 = z2,
    x1 = x1, x2 = x2,
    death, pid)
  ret
}

dat1 <- genData()

## create a dataset with repeated measurements x
dat2 <- dat1 %>% pivot_longer(c(x1, x2), names_to='visit', values_to='x') %>%
  dplyr::select(x, trt, visit, pid) %>% as.data.frame()

dat2$visit <- as.factor(dat2$visit)
dat2$pid <- as.factor(dat2$pid)

fit <- jointCovariance(
  coxph_(Surv(time = y, event = death) ~ trt, data_index = 1),
  logrank_(Surv(time = y, event=death) ~ trt, data_index = 1),
  glm_(z1 ~ trt, family = 'binomial', data_index = 1),
  glm_(z2 ~ trt, family = 'gaussian', data_index = 1),
  mmrm_(x ~ trt + us(visit | pid), reml = TRUE, data_index = 2),
  gee_(x ~ trt, family = 'gaussian', corstr = 'independence', data_index = 2),
  data = list(dat1, dat2))

fit


bfit <-jointCovariance(
  coxph_(Surv(time=y, event=death) ~ trt, data_index = 1),
  logrank_(Surv(time=y, event=death) ~ trt, data_index = 1),
  glm_(z1 ~ trt, family = 'binomial', data_index = 1),
```

```
  glm_(z2 ~ trt, family = 'gaussian', data_index = 1),
  mmrm_(x ~ trt + us(visit | pid), reml = TRUE, data_index = 2),
  gee_(x ~ trt, family = 'gaussian', corstr = 'independence', data_index = 2),
  data = list(dat1, dat2), nboot = 10)

summary(bfit)

## km_() and quantile_() require nboot > 0 because they have no
## closed-form score. compute_cov is forced to FALSE for km_().
## When all models share one dataset, `data_index` and `list(...)`
## can be omitted.
kfit <- jointCovariance(
  km_(Surv(time = y, event = death) ~ trt, conf_type = 'log',
      times = c(0.5, 1, 1.5)),
  glm_(z1 ~ trt, family = 'binomial'),
  data = dat1, nboot = 30, seed = 1)

qfit <- jointCovariance(
  quantile_(y ~ trt, probs = c(0.25, 0.5, 0.75)),
  glm_(z2 ~ trt, family = 'gaussian'),
  data = dat1, nboot = 30, seed = 1)
```

`km_` *Creating Objects of Kaplan-Meier Curve*

Description

`km_` is a wrapper function creating an object of Kaplan-Meier curve to be passed into `jointCovariance`, the main function of this package through its argument `...`. The object defines how a Kaplan-Meier curve would be fitted.

Usage

```
km_(formula, times = NULL, conf_type, data_index = 1)
```

Arguments

`formula`	a formula created by `survival::Surv()`.
`times`	numeric vector of time. Survival probabilities at `times` are computed (with g-transformation defined by `conf_type`).
`conf_type`	character. Type of confidence interval. It must be one of `"log"`, `"log-log"`, `"plain"`, `"logit"`, or `"arcsin"`.
`data_index`	integer. Index of the data frame in the `data` argument of `jointCovariance` to be used when fitting a generalized linear model.

Details

Usually, g-transformation is applied to the survival probability `S(t)` to obtain pointwise confidence interval of a Kaplan-Meier curve. This can be achieved by specifying `conf_type`. For identity transformation, use `conf_type = "plain"`.

This function can only be used with `jointCovariance` when the bootstrap method is used to estimate variance-covariance matrix of multiple outcome models.

`logrank_` *Creating Objects of Logrank Test*

Description

`logrank_` is a wrapper function `survival::coxph` to create an object to be passed into `jointCovariance`, the main function of this package through its argument `...`. Logrank test is the score test under the proportional hazards regression model. The object defines how a logrank test would be computed.

Usage

```
logrank_(formula, ties = c("efron", "breslow", "exact"), data_index = 1)
```

Arguments

`formula`	see `formula` in `survival::coxph`.
`ties`	character string specifying the method for tie handling. One of `"efron"` (default), `"breslow"`, or `"exact"`. Passed through to `survival::coxph`.
`data_index`	integer. Index of the data frame in the `data` argument of `jointCovariance` to be used when computing testing statistic of logrank test.

Details

Not all arguments of `survival::coxph` are supported in `logrank_` due to the complexity in handling environment and scope, which is particularly difficult for arguments like `weights`, `subset`, etc.

`mmrm_` *Creating Objects of Mixed Models for Repeated Measures*

Description

`mmrm_` is a wrapper function of `mmrm::mmrm` to create an object to be passed into `jointCovariance`, the main function of this package through its argument `...`. The object defines how a MMRM model would be fitted.

Usage

```
mmrm_(
  formula,
  covariance = NULL,
  reml = TRUE,
  control = mmrm::mmrm_control(...),
  ...,
  data_index = 1
)
```

Arguments

`formula`	see `formula` in `mmrm::mmrm`.
`covariance`	see `covariance` in `mmrm::mmrm`.
`reml`	see `reml` in `mmrm::mmrm`.
`control`	see `control` in `mmrm::mmrm`.
`...`	see `...` in `mmrm::mmrm`.
`data_index`	integer. Index of the data frame in the `data` argument of `jointCovariance` to be used when fitting a GEE model.

Details

The argument `weights` of `mmrm::mmrm` is supported in `mmrm_` due to the complexity in handling environment and scope.

Please always refer to help document of `mmrm::mmrm` before using `mmrm_`. For example, time variable and observation ID must be factor variables in some cases, otherwise error may be prompted. Users can call `mmrm::mmrm` using the same arguments being passed to `mmrm_` to check validity.

`pated` *Prognostic Variables Assisted Treatment Effect Detection*

Description

`pated` is a wrapper function of `jointCovariance` for testing treatment effect in randomized clinical trials. It assumes that prognostic variables are fully randomized. This assumption can help enhancing statistical power of conventional approaches in detecting the treatment effect. Specifically, the sensitivity of the conventional models specified in `...` are improved by `pated`.

Usage

```
pated(
  ...,
  data,
  nboot = 0,
  compute_cov = TRUE,
```

```
  seed = NULL,
  transform = "identity"
)
```

Arguments

`...`	model specifications built by `glm_()`, `coxph_()`, `logrank_()`, `gee_()`, `mmrm_()`, `km_()`, or `quantile_()`. The first specification is the primary outcome whose treatment effect is being tested; the rest are prognostic covariates used to tighten the SE.
`data`	either a single data frame (when all models are fitted on the same dataset) or a list of data frames (one entry per `data_index`). Each data frame must have a `pid` column carrying subject identifiers; records with the same `pid` across different data frames refer to the same subject.
`nboot`	non-zero integer if bootstrap is adopted. By default 0.
`compute_cov`	logic. If `TRUE`, empirical covariance matrix is computed using bootstrap estimate and returned. Bootstrap estimate will be abandoned. If `FALSE`, bootstrap estimate will be returned and no empirical covariance matrix is computed.
`seed`	random seed when generate bootstrap data.
`transform`	character. Now only supports `"identity"`.

Value

a data frame of testing results.

Examples

```
## More examples can be found in the vignettes.
library(survival)
library(mvtnorm)

genData <- function(seed = NULL){
  set.seed(seed)
  n <- 300
  sigma <- matrix(c(1, .6, .6, 1), 2)
  x <- rmvnorm(n, sigma = sigma)
  z1 <- rbinom(n, 1, .6)
  z2 <- rnorm(n)
  trt <- rbinom(n, 1, .5)

  bet <- c(-.2, .2)
  y <- -.5 + x %*% bet + z1 * .3 - z2 * .1 + .1 * trt - .1 * rnorm(n)
  death <- rbinom(n, 1, .8)
  data.frame(
    y = as.numeric(y), trt = trt,
    z1 = z1, z2 = z2,
    x1 = x[, 1], x2 = x[, 2],
    death, pid = paste0('s-', seq_len(n))
  )
}
```

```
dat <- genData(seed = 31415926)

## `data_index` defaults to 1 in every spec constructor and a single
## data.frame is auto-wrapped into a list, so neither needs spelling out
## when all models are fitted on the same dataset.
fit <-
  pated(
    coxph_(Surv(time = y, event = death) ~ trt),
    glm_(z1 ~ trt, family = 'binomial'),
    glm_(z2 ~ trt, family = 'gaussian'),
    glm_(x1 ~ trt, family = 'gaussian'),
    glm_(x2 ~ trt, family = 'gaussian'),
    data = dat
  )

fit
```

`plot.pated` *Plot PATED Analysis Results*

Description

Plot PATED Analysis Results

Usage

```
## S3 method for class 'pated'
plot(x, ...)
```

Arguments

`x`	an object returned from `pated()`.
`...`	currently not supported.

Value

`NULL`

`print.summary.jointCovariance` *Title Summarize an Analysis of Multiple Outcomes.*

Description

Summarize an analysis of multiple outcomes.

Usage

```
## S3 method for class 'summary.jointCovariance'
print(x, ...)
```

Arguments

`x`	an object returned by `jointCovariance()`.
`...`	for debugging only.

Value

an invisible object.

Examples

```
## no example
```

`quantile_` *Creating Objects of Group Quantile Differences*

Description

`quantile_` is a wrapper function creating an object that, for each requested probability, computes the difference between the two arms' sample quantiles of the outcome. The object is passed to `jointCovariance` or `pated` through `...`. Because the empirical-quantile estimator has no tractable closed-form score, this engine is available only through the bootstrap path (`nboot` > `0`).

Usage

```
quantile_(formula, probs = c(0.25, 0.5, 0.75), data_index = 1)
```

Arguments

`formula`	a two-sided formula `y ~ arm` where the right-hand side is a binary grouping variable.
`probs`	numeric vector of probabilities in `(0, 1)`. By default the first quartile, median, and third quartile.
`data_index`	integer. Index of the data frame in the `data` argument of `jointCovariance` to be used.

`simulateMoData` *Generating Data for Simulation and Testing*

Description

`simulateMoData` generates data for simulation and testing purposes.

Usage

```
simulateMoData(n = 500, hr = 0.8, seed = NULL)
```

Arguments

`n`	an integer for total sample size of a randomized control trial of two arms.
`hr`	hazard ratio of treatment.
`seed`	random seed. By default `NULL` for no seed being specified.

`summary.jointCovariance` *Object Summaries*

Description

`summary` method for class `jointCovariance`.

Usage

```
## S3 method for class 'jointCovariance'
summary(object, model_index = NULL, ...)
```

Arguments

`object`	an object returned by `jointCovariance()`.
`model_index`	`NULL` if displaying summary of all fitted models; otherwise, an integer indicating the fitted model.
`...`	for debugging only

Value

a list

`vcov.jointCovariance` *Calculate Variance-Covariance Matrix for a Fitted Model Object*

Description

Returns the variance-covariance matrix of the main parameters of fitted model objects. The "main" parameters of models correspond to those returned by `coef`.

Usage

```
## S3 method for class 'jointCovariance'
vcov(object, model_index = NULL, ...)
```

Arguments

`object`	an object returned by `jointCovariance()`.
`model_index`	`NULL` if displaying covariance matrix of all fitted models; otherwise, an integer indicating the fitted model.
`...`	for debugging only

Value

a matrix of covariance of all estimates

Index